AD-A070 518 ARMY ARMAMENT RESEARCH AND DEVELOPMENT COMMAND DOVER--ETC F/G 12/1
THE HARR TRANSFORM: ITS THEORY AND COMPUTER IMPLEMENTATION.(U)
APR 79 G SIVAK

UNCLASSIFIED ARSCD-TR-78005 SBIE -AD-E400 309 NL

1 OF 1

AD
A070518

END
DATE
FILMED
8--79
DDC

12 LEVEL III

AD

AD-E400 309

AD A070518

TECHNICAL REPORT ARSCD-TR-78005

THE HAAR TRANSFORM:
ITS THEORY AND COMPUTER IMPLEMENTATION

GARY SIVAK

APRIL 1979

US ARMY ARMAMENT RESEARCH AND DEVELOPMENT COMMAND
FIRE CONTROL AND SMALL CALIBER
WEAPON SYSTEMS LABORATORY
DOVER, NEW JERSEY

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

79 06 01 018

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER Technical Report ARSCD-TR-78005	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) THE HAAR TRANSFORM: ITS THEORY AND COMPUTER IMPLEMENTATION		5. TYPE OF REPORT & PERIOD COVERED FINAL 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Gary Sivak		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS ARRADCOM, FC&SCWSL ATTN: DRDAR-SCF-IE Dover, NJ 07801		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 61101A, 1T161101A91A 00, 243 F6
11. CONTROLLING OFFICE NAME AND ADDRESS ARRADCOM, TSD ATTN: DRDAR-TSS, STINFO Div Dover, NJ 07801		12. REPORT DATE APRIL 1979 13. NUMBER OF PAGES 57
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) ARRADCOM, FC&SCWSL ATTN: DRDAR-SCF-IE Dover, NJ 07801		15. SECURITY CLASS. (of this report) UNCLASSIFIED 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

393 302

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Haar Transform
Haar Functions
Fast Haar Transform Digital Data Analysis
Transformations (Mathematics)
Digital Image Processing

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

Haar functions, square wave step functions of various rates of value change or "sequency", take on the values +1, -1, and 0 in the interval (0, 1). Since Haar functions form an orthogonal system, any digital function or set of digital data can be represented as a linear combination of these functions, just as Fourier analysis represents data as a linear combination of sines and cosines.

→ next page

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

20. (Cont'd)

The Fast Haar Transform algorithm developed in this paper drastically reduces the number of operations required to transform a set of 16 data elements, i.e., from 256 to only 30. Therefore, one of the chief advantages that the Fast Haar Transform has over the Fast Fourier Transform is that it is between four and five times faster in terms of the number of computer calculations required, thereby reducing the cost of the computer time needed by 80 percent.

ACKNOWLEDGMENTS

The author wishes to thank Messrs. Thomas Gavanis, Bernard Peskin, and Howard Jenkinson of Fire Control for their valuable suggestions on how to improve this report for maximum clarity. Appreciation is also extended to Mr. Forrest McMains of MISD for introducing the author to Xedit, the computerized text-editing system, which proved invaluable in the preparation of this document.

TABLE OF CONTENTS

TABLES

FIGURES

INTRODUCTION

This report documents research in the development of a Fast Haar Transform algorithm and its application to digital data analysis, as performed by the author during the period June-to-October of 1976. It is an outgrowth of previous research in the area of transform application to image processing.

Just as the Fourier Transform represents data as a linear combination of sines and cosines, the Haar Transform represents a function as a linear combination of square-wave step functions called Haar functions. These functions are defined and grouped into matrices from which a transform is developed. Then the fast transform algorithm is presented with a flowchart and examples. A summary of the advantages that the Fast Haar Transform (FHT) has over the Fast Fourier Transform (FFT) is presented at the end of the report.

THEORY OF HAAR FUNCTIONS

It is instructive to examine the Haar functions, to order them into matrices, and to formulate a fast transform algorithm that can be computer-implemented for digital data analysis.

Around 1900, Haar defined the set of discontinuous step functions which are used to obtain the transform that today bears his name (ref 1). The Haar functions (step functions of various rates of value change or "sequency") take on the values +1, 0, and -1 in the closed interval,

$$[0, 1] \text{ i.e. } 0 \leq x \leq 1$$

The first and simplest Haar function is:

$$H_0^0(x) = 1, \qquad x \in [0, 1] \tag{1}$$

The second Haar function is:

$$H_0^1(x) = \begin{cases} 1 & x \in [0, 1/2) \\ 0 & x = 1/2 \\ -1 & x \in (1/2, 1] \end{cases} \tag{2}$$

Finally, in general, for $m \geq 1$, and $1 \leq k \leq 2^m$

$$
H_0^1(x) = \begin{cases} \sqrt{2^m} & x \in \left(\frac{k-1}{2^m}, \frac{k-\frac{1}{2}}{2^m}\right) \\ -\sqrt{2^m} & x \in \left(\frac{k-\frac{1}{2}}{2^m}, \frac{k}{2^m}\right) \\ 0 & x \in \left(\frac{\ell-1}{2^m}, \frac{\ell}{2^m}\right) \end{cases} \tag{3}
$$

for $\ell \neq k$ and $1 \leq \ell \leq 2^m$

By choosing any integral value of m, one can vary k and construct any of the 2^m Haar functions of the m^{th} order desired.

At points of discontinuity, let $H_m^k(x)$ be the mean on either side of adjoining intervals, i.e., at the points 0, and 1, let $H_m^k(x)$ take on its values as in the intervals $(0, \frac{1}{2^{m+1}})$ and $(1- \frac{1}{2^{m+1}}, 1)$, respectively.

The total collection of the Haar functions is a complete set and an orthonormal system, that is one in which the functions are normalized and orthogonal. Clearly, $H_m^k(x)$ is normalized. Also, the $H_o^k(x)$ functions are orthogonal to all others, that is, the integral over the domain of definition of their product vanishes. Finally, in general, for $m \geq 1$ and $1 \leq i, j \leq 2^m$, for $i \neq j$:

$$
\int_0^1 H_m^i(x)\, H_m^j(x)\, dx = 0 \tag{4}
$$

while for $n > m$, the interval in which $H_n^i(x)$ does not vanish is contained in an interval of constant length of $H_m^j(x)$ and therefore:

$$\int_0^1 H_n^i(x)\, H_m^j(x)\, dx = \pm \sqrt{2}^{m} \int_0^1 H_n^i(x)\, dx = 0 \qquad (5)$$

Thus, the existence of an orthonormal set of functions is established.

MATRICES AND BASIS OF TRANSFORM

The fact of orthogonality is important and means, therefore, that a function can be expressed in terms of or represented by a linear combination of Haar functions. This means that for a given digital function, $F_d(x_i)$, for i = 0, 1, 2 ..., N-1, that is, over the range of the index i for which $F_d(x_i)$ is valid, one can write a Haar function decomposition as follows:

$$F_d(x_i) = \sum_{j=1}^{m} \sum_{k=1}^{2j} c_{jk} H_j^k(x_i) \qquad (6)$$

where j represents the particular order of the given Haar function under consideration of which 2^j or k individual cases exist. The quantity C_{jk} is the weighting factor, the real number designating how much of a contribution $H_j^k(x_i)$ is giving to the digital function or data string series $F_d(x_i)$.

In general, for functions in the complex plane, one simply treats the real and imaginary parts separately and writes:

$$F_d(x_i) = R_d(x_i) + jI_d(x_i) \qquad (7)$$

where $j = \sqrt{-1}$ and $R_d(x_i)$ and $I_d(x_i)$ are the real and imaginary parts that are expanded in terms of Haar functions as in equation 6.

Thus, any set of digital data can be expressed in terms of Haar function weighting coefficients, just as N data points can be expressed as a linear combination of sines and cosines in the Fourier Transform representation.

For Haar functions, a transform is most easily motivated by arranging the functions in order of increasing sequency, or frequency of value change, into one or another of the N by N Haar matrices with:

$$N = 2^n \tag{8}$$

where n is a positive integer and where N and n are analogous to k and m, respectively, of equation 6.

Note, in equation 3, the presence of the bothersome square roots $\pm \sqrt{2^m}$ in the Haar functions and also in the matrix elements. If the Haar Transform is represented by an N by N matrix T and its inverse by T^I , the following condition is desired:

$$T\,T^I = T^I T = I_N \tag{9}$$

where I_N is the N by N identity matrix with 1's on the main diagonal and 0's everywhere else. The factors $\pm \sqrt{2^m}$, from both the Haar transform matrix and its inverse, can be grouped onto the inverse; therefore, for a given order of Haar functions, simply divide by $N = 2^n$. Thus modified, the Haar Transform matrix for the N = 16 case is given in figure 1. Its inverse, except for the outside factor of 1/N, is given in figure 2, which was computer generated by program HIMAT. (See appendix A.)

To perform actual processing, take an input data string with elements IN_j, the Haar Transform matrix with elements T_{ij} with the inverse transform matrix T^I_{ki}, the output data string with elements OUT_k, and a storage string with elements X_i, where, i and j range from 1 to N. Multiply the input data by the transform matrix, proceeding down the rows and storing the sums in X. Multiply X by the inverse transform matrix, proceeding down the rows and storing the results in the output data string. In equation form:

$$X_i = \sum_{j=1}^{N} T_{ij}\, IN_j \qquad OUT_k = \sum_{i=1}^{N} T^I_{ki}\, X_i \tag{10}$$

where in T_{ij} and T^I_{ki}, the first index designates the row, and the second the column position for the given matrix element.

$$
\begin{bmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \\
1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\
1 & 1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & -1 & -1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & -1 & -1 \\
1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1
\end{bmatrix}
$$

Figure 1. The Haar 16-by-16 transform matrix.

$$
\begin{bmatrix}
1 & 1 & 2 & 0 & 4 & 0 & 0 & 0 & 8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 2 & 0 & 4 & 0 & 0 & 0 & -8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 2 & 0 & -4 & 0 & 0 & 0 & 0 & 8 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 2 & 0 & -4 & 0 & 0 & 0 & 0 & -8 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & -2 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 8 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & -2 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & -8 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & -2 & 0 & 0 & -4 & 0 & 0 & 0 & 0 & 0 & 8 & 0 & 0 & 0 & 0 \\
1 & 1 & -2 & 0 & 0 & -4 & 0 & 0 & 0 & 0 & 0 & -8 & 0 & 0 & 0 & 0 \\
1 & -1 & 0 & 2 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 8 & 0 & 0 & 0 \\
1 & -1 & 0 & 2 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & -8 & 0 & 0 & 0 \\
1 & -1 & 0 & 2 & 0 & 0 & -4 & 0 & 0 & 0 & 0 & 0 & 0 & 8 & 0 & 0 \\
1 & -1 & 0 & 2 & 0 & 0 & -4 & 0 & 0 & 0 & 0 & 0 & 0 & -8 & 0 & 0 \\
1 & -1 & 0 & -2 & 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 8 & 0 \\
1 & -1 & 0 & -2 & 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & -8 & 0 \\
1 & -1 & 0 & -2 & 0 & 0 & 0 & -4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 8 \\
1 & -1 & 0 & -2 & 0 & 0 & 0 & -4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -8
\end{bmatrix}
$$

Figure 2. The inverse Haar Transform matrix, N = 16 case.

During processing, the Haar Transform matrix operates on each element of the data string, sampling at skip frequencies differing by powers of 2 from coarse to fine. The lowest is a DC term, the highest being value changes between adjacent data elements, i.e., the bandwidth limit of the digital input. Specifically, for the 16 by 16 transform matrix case, the DC level and spatial frequency alternations of 1, 2, 4, and 8 cycles per the data field length of 16 points occur. In this case, the Haar Transform represents each data string of equation 6 as equation 11:

$$F_d\ (x_i) = F_8 + F_4 + F_2 + F_1 + \text{DC term} \tag{11}$$

where F_8, F_4, F_1, and F , are square wave spatial frequency contributions of 8, 4, 2, and 1 cycle per data field length, respectively. (See ref 2.) These are square wave functions that take on the values $\pm a$, $\pm b$, $\pm c$, $\pm d$, and e, for the DC term as shown in figure 3.

Equation 11 is a restatement of equation 6, and shows how a data set is constructed from its individual Haar components.

A demonstration of orthogonality is instructive to show the existence of the Haar components in equation 11, by inverting it to obtain F_8, F_4, F_2, and F_1 as functions of the input data points $F_d\ (x_i)$. One therefore writes out the Haar component contributions in equation 11 as specified in figure 3 for all 16 digital data points $F_d\ (x_i)$

$$\begin{array}{lcl}
F_8 & = & -a+a-a+a-a+a-a+a-a+a-a+a-a+a-a+a \\
F_4 & = & -b-b+b+b-b-b+b+b-b-b+b+b-b-b+b+b \\
F_2 & = & -c-c-c-c+c+c+c+c-c-c-c-c+c+c+c+c \\
F_1 & = & -d-d-d-d-d-d-d-d+d+d+d+d+d+d+d+d \\
DC & = & e\ e\ e\ e\ e\ e\ e\ e\ e\ e\ e\ e\ e\ e\ e\ e \\
P_i & = & 1\ 2\ 3\ 4\ 5\ 6\ 7\ 8\ 9\ 10\ 11\ 12\ 13\ 14\ 15\ 16
\end{array} \tag{12}$$

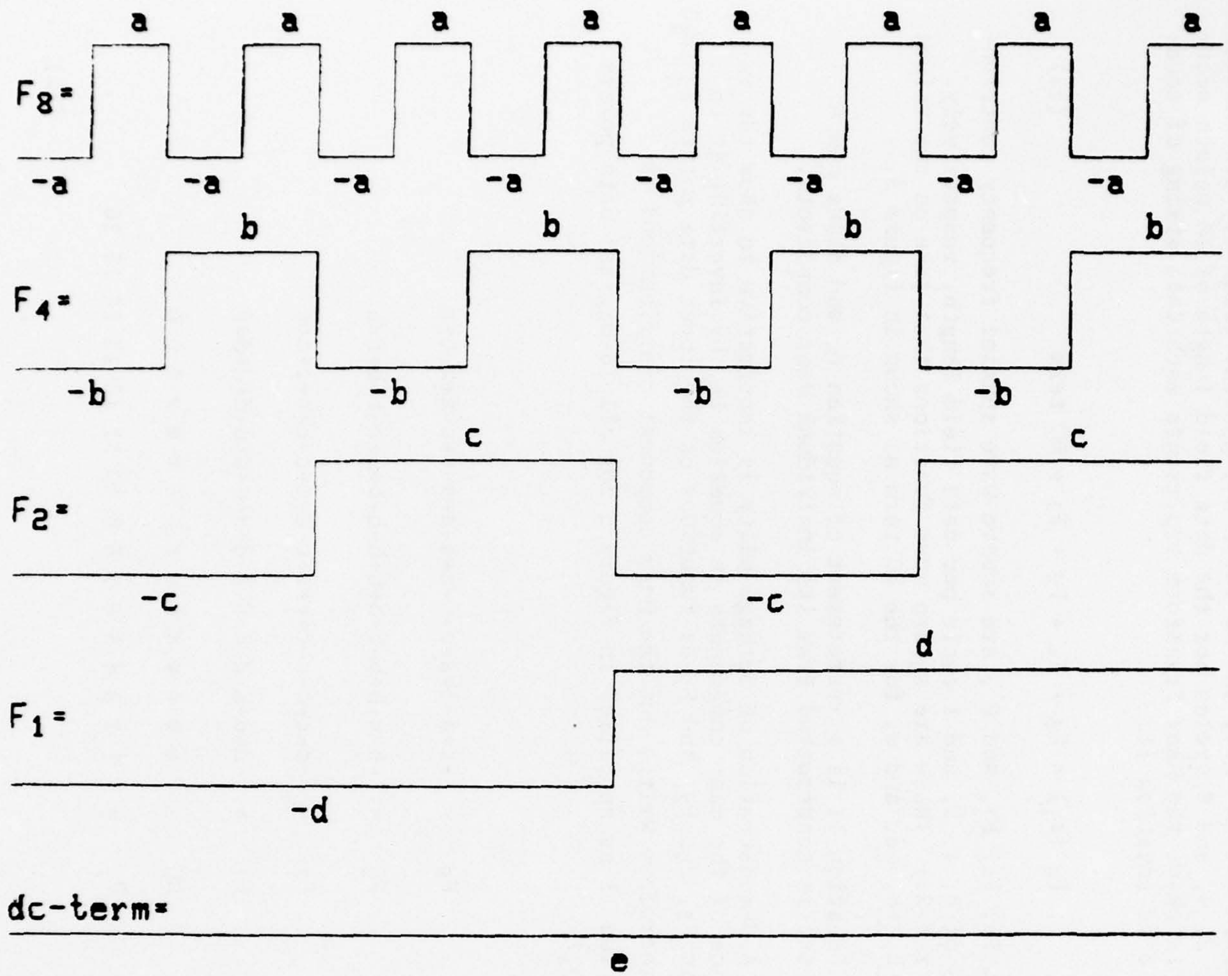

Figure 3. Square wave patterns for Haar Transform.

where P_i is the data point in question. One adds the corresponding inputs for each of the data points P_i in $F_d(x_i)$ to get:

$$P_1 = -a -b -c -d +e \qquad P_2 = a -b -c -d +e$$

$$P_3 = -a +b -c -d +e \qquad P_4 = a +b -c -d +e$$

$$P_5 = -a -b +c -d +e \qquad P_6 = a -b +c -d +e$$

$$P_7 = -a +b +c -d +e \qquad P_8 = a +b +c -d +e$$

$$P_9 = -a -b -c +d +e \qquad P_{10} = a -b -c +d +e$$

$$P_{11} = -a +b -c +d +e \qquad P_{12} = a +b -c +d +e$$

$$P_{13} = -a -b +c +d +e \qquad P_{14} = a -b +c +d +e$$

$$P_{15} = -a +b +c +d +e \qquad P_{16} = a +b +c +d +e \tag{13}$$

Let:

$$x_1 = a$$

$$x_2 = b$$

$$x_3 = c$$

$$x_4 = d \tag{14}$$

$$x_5 = e$$

With 5 inputs, this system of equations determines 16 outputs. Thus, to invert it, one has the choice of which outputs to use. Try:

$$y_1 = P_{16} = x_1 + x_2 + x_3 + x_4 + x_5$$

$$y_2 = P_{14} = x_1 - x_2 + x_3 + x_4 + x_5$$

$$y_3 = P_{12} = x_1 + x_2 - x_3 + x_4 + x_5$$

$$y_4 = p_8 = x_1 + x_2 + x_3 - x_4 + x_5$$

$$y_5 = p_{10} = x_1 - x_2 - x_3 + x_4 + x_5$$

(15)

The matrix for this system is:

$$M = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & -1 & 1 \\ 1 & -1 & -1 & 1 & 1 \end{bmatrix} \qquad (16)$$

Notice that columns 1 and 5 are identical, the determinant is 0, and the system cannot be inverted. So let:

$$x_1 = x_1 + x_5$$

$$x_2 = x_2 , x_3 = x_3 , x_4 = x_4 , x_5 = x_5$$

$$y_1 = y_1 , y_2 = y_2 , y_3 = y_3 , y_4 = y_4 , y_5 = y_5$$

(17)

Then choose the equations for y_1 through y_4 and write the matrix for the 4-by- system:

$$M = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & 1 \\ 1 & 1 & -1 & 1 \\ 1 & 1 & 1 & -1 \end{bmatrix} \tag{18}$$

As found by computer generation, the inverse of the matrix is:

$$M^{I} = \begin{bmatrix} -.5 & .5 & .5 & .5 \\ .5 & -.5 & 0 & 0 \\ .5 & 0 & -.5 & 0 \\ .5 & 0 & 0 & -.5 \end{bmatrix} \tag{19}$$

Thus, one obtains equation 20:

$$\begin{aligned} X_1 &= (-Y_1 + Y_2 + Y_3 + Y_4)/2 \\ x_2 &= (y_1 - y_2)/2 \\ x_3 &= (y_1 - y_3)/2 \\ x_4 &= (y_1 - y_4)/2 \end{aligned} \tag{20}$$

and therefore from equations 14, 15, and 17:

$$\begin{aligned} a + e &= (p_{14} + p_{12} + p_8 - p_{16})/2 \\ b &= (p_{16} - p_{14})/2 \\ c &= (p_{16} - p_{12})/2 \\ d &= (p_{16} - p_8)/2 \end{aligned} \tag{21}$$

Note that not all square wave components in equation 21 are obtained as independent expressions, i.e., only the sum of a and e is present. The reason is that the signs of a, b, c, and d have not been specified. This is equivalent to not having specified the phases of the contributing square wave oscillations. An ambiguity of a 180-degree phase angle in each one occurs because a statement has not been made as to whether the oscillation should begin positive or negative. The Haar transform, however, does this, and, therefore, accurately represents the original input data.

Comparing figure 1 with equation 10, note that for the transform data storage X, X_1 corresponds to the matrix elements in group 1 of the figure since only these elements were used to compute it. Similarly, X_2 corresponds to group 2, X_3 and X_4 to group 3, X_5 through X_8 to group 4, X_9 through X_{16} to group 5. The magnitude of the elements in group 1 of the Haar Transform matrix affect the strength of the DC term. Similarly, the magnitude of the elements in groups 2, 3, 4, and 5 affect the relative weights of the Haar contributions of F_1, F_2 F_4 and F_8 respectively, of equation 11. That is, for $1 \leq i \leq 5$, multiplying the elements of group i by an enhancement factor $m_i > 1$ will enhance the presence of the i^{th} spatial frequency component. Conversely, incorporating factors $m_i < 1$ will suppress the presence of the given square wave spatial frequency pattern. This has applications to bandwidth reduction and noise-suppression capability.

THE FAST HAAR TRANSFORM

Motivation

Now that the nature and effect of the Haar Transform are understood, working out the N = 16 case in detail will demonstrate the motivation for the algorithm obtained. The author-designed and tested Fast Haar Transform is more efficient for machine computation than is the indexing of a computer through the numerous rows and columns of extensive,memory-consuming N by N matrices.

The N = 16 case was found to be the optimum for illustrative purposes because the N = 32 case is too unwieldy to be written; every case smaller is too trivial and has too few details and operations to clearly demonstrate all the procedures involved.

As dictated by equation 10, one writes out the 16 components of the Haar Transform T as a function of the components of a digital input vector I. This matrix multiplication of each of 16 elements in

each of 16 rows requires 256 operations. By regrouping the computations into recursive patterns of accumulating sums and differences, one will obtain the same results in 30 operations. This computational reorganization is the Fast Haar Transform. The details follow:

$$T_1 = I_1 + I_2 + I_3 + I_4 + I_5 + I_6 +$$
$$I_7 + I_8 + I_9 + I_{10} + I_{11} + I_{12} +$$
$$I_{13} + I_{14} + I_{15} + I_{16}$$

$$T_2 = I_1 + I_2 + I_3 + I_4 + I_5 + I_6 +$$
$$I_7 + I_8 - I_9 - I_{10} - I_{11} - I_{12} -$$
$$I_{13} - I_{14} - I_{15} - I_{16}$$

$$T_3 = I_1 + I_2 + I_3 + I_4 -$$
$$I_5 - I_6 - I_7 - I_8$$

$$T_4 = I_9 + I_{10} + I_{11} + I_{12} -$$
$$I_{13} - I_{14} - I_{15} - I_{16}$$

$$T_5 = I_1 + I_2 - I_3 - I_4$$

$$T_6 = I_5 + I_6 - I_7 - I_8$$

$$T_7 = I_9 + I_{10} - I_{11} - I_{12}$$

$$T_8 = I_{13} + I_{14} - I_{15} - I_{16}$$

$$T_9 = I_1 - I_2$$

$$T_{10} = I_3 - I_4$$

$$T_{11} = I_5 - I_6$$

$$T_{12} = I_7 - I_8$$

$$T_{13} = I_9 - I_{10}$$

$$T_{14} = I_{11} - I_{12}$$

$$T_{15} = I_{13} - I_{14}$$

$$T_{16} = I_{15} - I_{16} \qquad (22)$$

Working through the input data, one groups the components into the pattern of accumulating sums and differences into which they naturally fall as follows:

$$A_1 = I_1 + I_2 \qquad A_5 = I_9 + I_{10}$$

$$A_2 = I_3 + I_4 \qquad A_6 = I_{11} + I_{12}$$

$$A_3 = I_5 + I_6 \qquad A_7 = I_{13} + I_{14}$$

$$A_4 = I_7 + I_8 \qquad A_8 = I_{15} + I_{16} \tag{23}$$

One continues this procedure:

$$B_1 = A_1 + A_2 \qquad B_3 = A_5 + A_6$$

$$B_2 = A_3 + A_4 \qquad B_4 = A_7 + A_8 \tag{24}$$

And finally one has:

$$C_1 = B_1 + B_2 \qquad C_2 = B_3 + B_4 \tag{25}$$

Note that equation 22 states the last N/2 Haar components directly, as differences between adjacent signal elements. By comparing the results of equations 23, 24, and 25, with equation 22, one finds that the first N/2 Haar Transform components of the input signal can also be expressed as differences. They are now generated in a reverse order, the first Haar Transform components last as follows:

$$T_5 = A_1 - A_2 \qquad T_7 = A_5 - A_6$$

$$T_6 \quad A_3 - A_4 \qquad T_8 = A_7 - A_8 \tag{26}$$

Also:

$$T_3 = B_1 - B_2 \qquad T_4 = B_3 - B_4 \tag{27}$$

And finally:

$$T_1 = C_1 + C_2 \qquad T_2 = C_1 - C_2 \tag{28}$$

Thus the Haar Transform has been implemented in a fast form.

Diagramatic Representation

The tree diagram of figure 4 is a computer generated signal flow graph for the N = 16 case of the Haar Transform. (See the listing of Program FIG4, appendix B.) It shows the flow of operations in computing the Haar components T_1 through T_{16} from the digital input signal components I_1 through I_{16}. The elements are generated in reverse order in aggregates corresponding to the groups in figure 1.

To take the Fast Haar Transform, compute the last 8 or N/2 elements, i.e., T_9 through T_{16}, or $T_{(N/2+1)}$ through T_N directly by taking adjacent input signal component differences. Then, sum element pairs to form the A array, take differences between adjacent elements; thus, obtaining transform components T_5 through T_8 or $T_{(N/4+1)}$ through $T_{N/2}$. Then, sum the elements of the A array to form the B's and take adjacent element differences to form components T_3 and T_4 or $T_{(N/8+1)}$ and $T_{N/4}$ Finally, sum again to form C_1 and C_2, and add and add and subtract to compute elements T_1 and T_2, respectively.

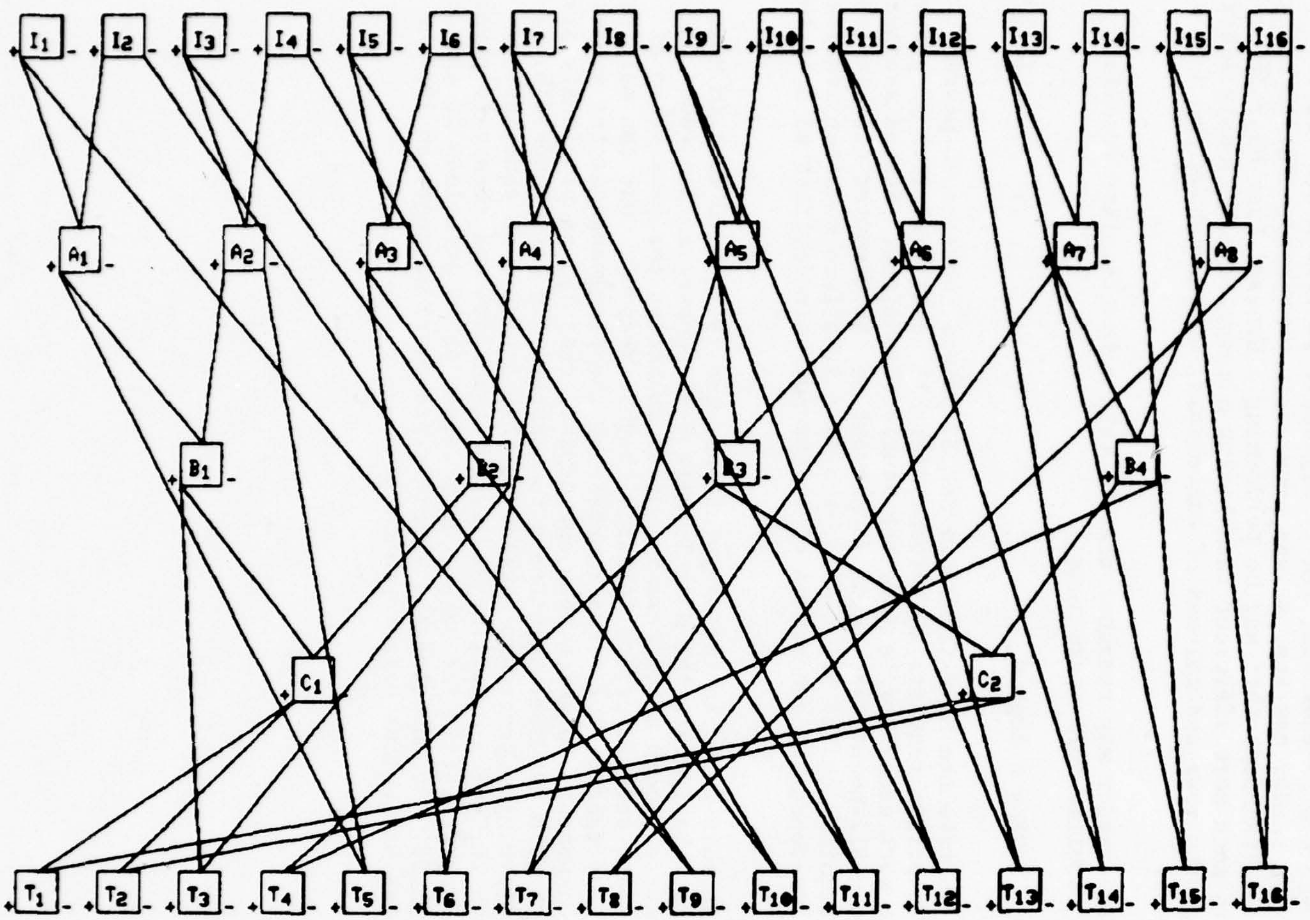

Figure 4. Haar Transform diagram, N = 16 case.

In taking the Haar Transform for the N = 16 case, note that 30 operations, 16 subtractions and 14 additions, are required. In general, for $N = 2^n$ elements, 2(N-1) algebraic additions are required. Thus, the reader can see, as was previously mentioned, that to obtain the Haar Transform this way be performing a General Fast Haar Transform is much more efficient than having a computer laboriously index through the rows and columns of cumbersome, memory-consuming, N by N matrices.

The reader may remember that to take a Fast Fourier Transform (FFT) requires more operations:

$$OP = N \log_2 N \tag{29}$$

Thus, to take the Haar Transform for 32 points requires 30 percent, for 128 points requires 29 percent, for 512 points requires 22 percent of the work necessary to take the Fourier Transform. Stated another way, the 512-point case saves 78 percent of the computer calculation time required by the FFT. Also, the Haar Transform does not require the time-consuming generation of trigonometric quantities as does the FFT.

At this point in the computations, one is free to work in Haar space and manipulate the Haar Transform components as desired. One can multiply some by factors > 1, thus enhancing the prominence of the square-wave functions to which they correspond. One can multiply others by factors < 1, thus suppressing the prominence of the square-wave components to which they correspond. Lastly, as previously stated, one can zero out some components to compress the data bandwidth. By performing a procedure similar to that employed in equations 22 through 28, one can develop an inverse transform and return to physical space to obtain the output data values as in equations 22 and 26 through 28. By adopting the specific procedure of equation 10, one computes the output data components:

$$O_1 = T_1 + T_2 + 2T_3 + 4T_5 + 8T_9$$

$$O_2 = T_1 + T_2 + 2T_3 + 4T_5 - 8T_9$$

$$O_3 = T_1 + T_2 + 2T_3 - 4T_5 + 8T_{10}$$

$$O_4 = T_1 + T_2 + 2T_3 - 4T_5 - 8T_{10}$$

$$O_5 = T_1 + T_2 - 2T_3 + 4T_6 + 8T_{11}$$

$$O_6 = T_1 + T_2 - 2T_3 + 4T_6 - 8T_{11}$$

$$O_7 = T_1 + T_2 - 2T_3 - 4T_6 + 8T_{12}$$

$$O_8 = T_1 + T_2 - 2T_3 - 4T_6 - 8T_{12}$$

$$O_9 = T_1 - T_2 + 2T_4 + 4T_7 + 8T_{13}$$

$$O_{10} = T_1 - T_2 + 2T_4 + 4T_7 - 8T_{13}$$

$$O_{11} = T_1 - T_2 + 2T_4 - 4T_7 + 8T_{14}$$

$$O_{12} = T_1 - T_2 + 2T_4 - 4T_7 - 8T_{14}$$

$$O_{13} = T_1 - T_2 - 2T_4 + 4T_8 + 8T_{15}$$

$$O_{14} = T_1 - T_2 - 2T_4 + 4T_8 - 8T_{15}$$

$$O_{15} = T_1 - T_2 - 2T_4 - 4T_8 + 8T_{16}$$

$$O_{16} = T_1 - T_2 - 2T_4 - 4T_8 - 8T_{16} \tag{30}$$

Note that to be properly scaled, these output results must be divided by a factor of N.

In a manner similar to equations 23 through 25, one forms groups of partial sums as intermediate steps in the computation of the output of equation 30 as follows:

$$A_1 = T_1 + T_2 \qquad A_2 = T_1 - T_2 \tag{31}$$

$$\begin{aligned} B_1 &= A_1 + 2T_3 & B_3 &= A_2 + 2T_4 \\ B_2 &= A_1 - 2T_3 & B_4 &= A_2 - 2T_4 \end{aligned} \tag{32}$$

$$\begin{aligned} C_1 &= B_1 + 4T_5 & C_5 &= B_3 + 4T_7 \\ C_2 &= B_1 - 4T_5 & C_6 &= B_3 - 4T_7 \\ C_3 &= B_2 + 4T_6 & C_7 &= B_4 + 4T_8 \\ C_4 &= B_2 - 4T_6 & C_8 &= B_4 - 4T_8 \end{aligned} \tag{33}$$

Finally, one can construct the output data as follows:

$$
\begin{aligned}
O_1 &= C_1 + 8T_9 & O_9 &= C_5 + 8T_{13} \\
O_2 &= C_1 - 8T_9 & O_{10} &= C_5 - 8T_{13} \\
O_3 &= C_2 + 8T_{10} & O_{11} &= C_6 + 8T_{14} \\
O_4 &= C_2 - 8T_{10} & O_{12} &= C_6 - 8T_{14} \\
O_5 &= C_3 + 8T_{11} & O_{13} &= C_7 + 8T_{15} \\
O_6 &= C_3 - 8T_{11} & O_{14} &= C_7 - 8T_{15} \\
O_7 &= C_4 + 8T_{12} & O_{15} &= C_8 + 8T_{16} \\
O_8 &= C_4 - 8T_{12} & O_{16} &= C_8 - 8T_{16}
\end{aligned}
\tag{34}
$$

As one can therefore see, the output components are computed from a recursive pattern of accumulating sums and differences. Haar components are added to the various A's, B's, and son on, according to the groups of figure 1 to which they correspond. The signs alternate in a regular fashion. As the index of the Haar components added to the sums of equations 31 through 34 runs from 1 through N, the factors that multiply them range from 1 through N/2.

Flowchart

This detailed analysis of the N = 16 case of the Haar Transform, although perhaps seemingly cumbersome, suggests a generalization of the FHT for N = 32, 64, or greater. The FHT is conveniently representable by a flowchart as in figure 5, that can be used to implement a computer routine for its calculation.

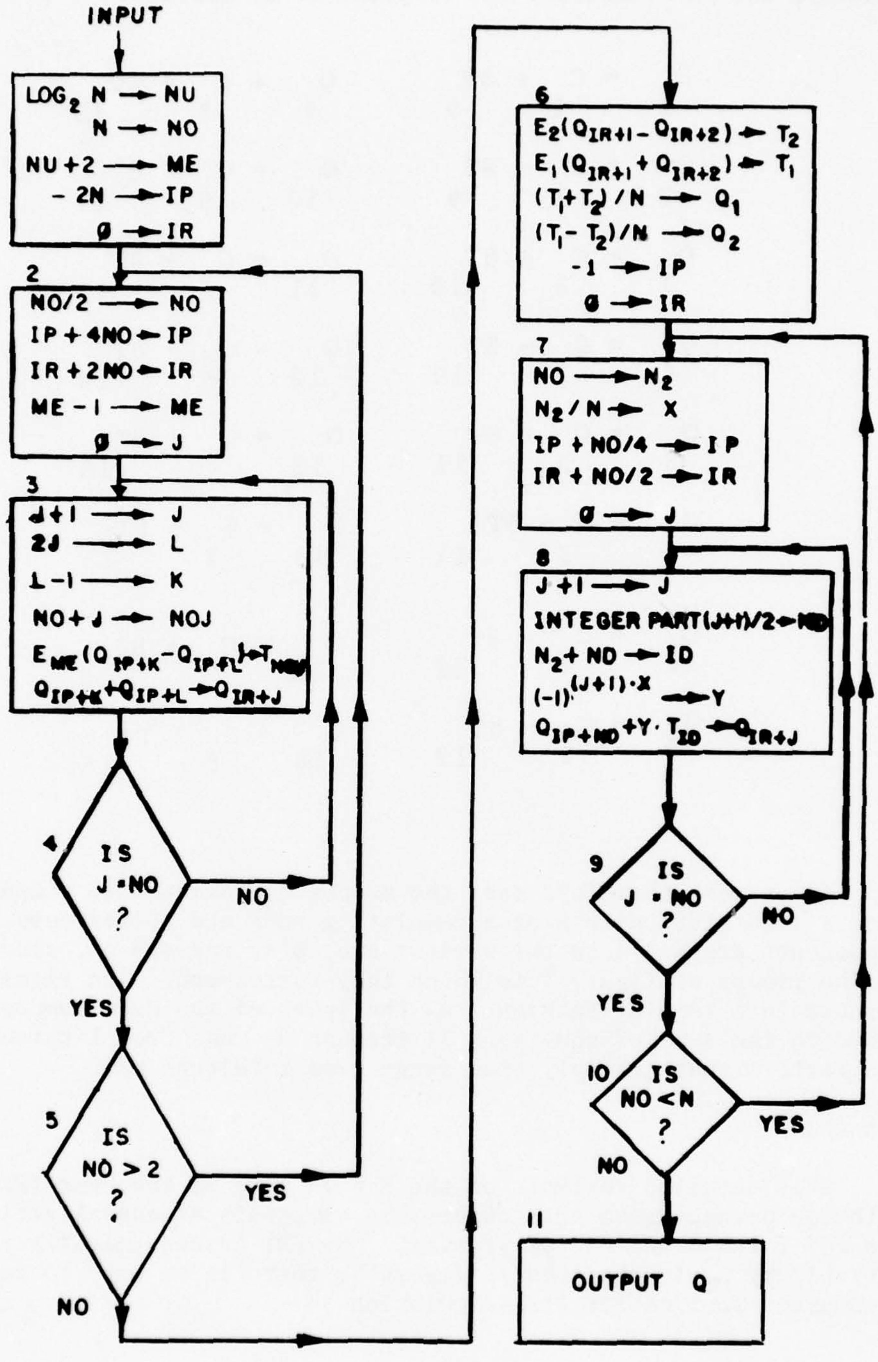

Figure 5. Fast Haar Transform flow chart.

In block 1, N is the number of points to transform, that must equal an integer power of 2. In block 2, IP and IR are the positions one counts from when desiring the current and soon-to-be computed values. The variable ME indicates which group of the transform one is currently in, to use to keep track of the E array, the enhancement or suppression multipliers. For N = 16, it is initially 5. In block 3, NOJ designates which transform component one currently wishes to compute. In the first pass, for the N = 16 case, it ranges from 9 through 16 as in the second half of equation 22, where one computes T_9 and Q_{IR+1}, which represents A_1. The indices L and K keep track of which transform components enter these computations. One tests for the end of the loop, and if it is reached, goes to block 5. If the variable NO > 2, one goes back to block 2, where the counters are reset, ME is decreased, and one is in the next group, i.e., 4. For the N = 16 case, one is now computing transform elements T_5 through T_8 and the B array.

When the loops break, one computes T_1 and T_2 directly. All this time, the multiplier factors are being factored into the transform components, doing all the data filtering and manipulation. Then in block 6, one computes Q_1 and Q_2, the A array of equation 31. Then, in block 7, NO is set to 4 and one computes the B array of equation 32. Finally, on the last pass, when NO = N, the output data points are computed as in equation 34. These values are the final answers and need not be scaled, for that was accomplished by the variable X in block 7. Block 11 contains 2N-2 entries, in the Q array, of which the last N are the outputs.

Fortran Subroutine

Table 1 is a listing of a FORTRAN subroutine which will implement the Haar Transform and return the output data of up through 64 points. The line numbers start with No. 610, to allow room for the placement of a main data-producing program ahead of the routine to form one file. The user is required to furnish the Q array of 126 or 2N-2 positions, the first 64 or N of which are the input data, while the last 62 may contain computer "garbage", since they will be overwritten by the partial sum arrays.

The user must also specify NU, the log to the base 2 of N, which for 64 points is of course 6. Giving a number smaller than this will implement the transform of correspondingly fewer points: 32, 16, etc. Finally, the user must furnish the array E, the NU+1 multipliers or enhancement factors that are incorporated into the transform to modify the data under analysis.

Table 1. FORTRAN subroutine to implement Haar Transform

```
      SUBROUTINE HAAR(Q,E,NU)
      DIMENSION T (64), Q (126), E (7)
      N =2**NU
      NO =N
      ME = NU+2
      IP = (-2)*N
      IR = 0
    5 NO = NO/2
      IP = IP + 4*NO
      IR = IR + 2*NO
      ME = ME-1
      DO 6 J =1,NO
      L = 2*J
      K = L-1
      NOJ = NO+J
      T(NOJ) = E(ME)*(Q(IP+K)-Q(IP+L))
    6 Q(IR+J) =Q(IP+K)+Q(IP+L)
      IF (NO.GT.2) GO TO 5
      T(2) =E(2)*(Q(IR+1)-Q(IR+2))
      T(1) =E(1)*(Q(IR+1)+Q(IR+2))
      Q(1) =(T(1)+T(2))/FLOAT(N)
      Q(2) =(T(1)-T(2))/FLOAT(N)
      IP= -1
      IR = 0
    7 N2 = NO
      X = FLOAT(N2)/FLOAT(N)
      NO = NO*2
      IP = IP + NO/4
      IR = IR + NO/2
      DO 8 J =1,NO
      ND = (J+1)/2
      ID = N2 + ND
      Y = (-1.0)**(J+1)*X
    8 Q(IR+J) = Q(IP+ND)+Y*T(ID)
      IF (NO.LT.N) GO TO 7
      RETURN
      END
```

APPLICATIONS

Signal Decomposition

The Haar Transform, besides being more efficient in terms of computer time than the Fast Fourier Transform, does not contain its inherent aliasing problem, which is that the last N/2 spatial frequency elements that the Fast Fourier Transform generates are completely superfluous data. By virtue of its construction, the highest spatial frequency that the Haar Transform algorithm samples is N/2 complete square-wave oscillations per N data points. For $NU = \log_2 N$, just NU+1 output weights exist, one for the DC term and one each for the NU Haar functions of the NU^{th} order being employed. Therefore, the Haar transform is capable of filtering and reconstructing any set of digital data, as has been shown, even if they are not explicitly periodic or continuous. The transform is widely applicable to the analysis, filtering, and bandwidth compression of any pulsed data. It can be employed for image processing and enhancement of video data (ref 2). Given a set of quantized video data, the Haar Transform can bring out edge and feature detail, i.e., improve the image contrast. It can suppress noise, i.e., increase the signal-to-noise ratio. Finally, it can be employed to decrease the amount of video data, i.e., compress the TV transmission bandwidth to speed up video data transmission, or alleviate the very likely possibility of channel crowding. Compression is done by zeroing out one or more of the highest square-wave components of portions of an image that contain little high-frequency information. Zeroing one component produces output with adjacent elements doubled, as in the digital output function:

$$OUT_D = 1,\ 1,\quad 2,\ 2,\quad 3,\ 3,\quad 2,\ 2,\quad 1,\ 1, \qquad (35)$$

One then selects every second point, cutting the data volume in half.

Table 1 is constructed from the output of computer program HARTST (app C). It presents an example of Haar Transform signal decomposition and shows the spatial frequency contributions to the nonperiodic digital ramp: 1, 2, 3, ... 16. This is the digital function:

$$F_d\ (x_i) \ = \ i + 1 \qquad (36)$$

where i = 0, 1, 2, ... 15. The table shows values ranging from the large DC term, through the last, the highest frequency of oscillation, i.e., 8 cycles in the data field length of 16 points.

Waveform Synthesis

Just as in the Fourier representation where one can construct a square-wave by the superpositioning of sine waves of designated frequency and amplitude, one can perfectly construct a sine wave by the superpositioning of its Haar Transform square-wave components.

This is demonstrated in the graphs of figures 6 through 12, where a 128-point, 1 cycle sine wave is approximated by 2, 3, 4, 5, 6, 7, and finally all of its Haar components as presented by Program FIG 6, (app D.) The figures are scaled to the same size. No DC-term is present, because the average value of sine x is, of course, 0.

The weights of the various Haar components are:

$$
\begin{aligned}
F_1 &= .63649 \\
F_2 &= -.01562 \\
F_4 &= -.27012 \\
F_8 &= -.18087 \\
F_{16} &= -.09643 \\
F_{32} &= -.04892 \\
F_{64} &= -.02453
\end{aligned}
\qquad (37)
$$

where the subscript on each variable is indicative of the number of cycles of oscillation of that given Haar component per data field length of 128 points.

Table 2. Square wave components of a Haar Transform signal decomposition

Input	DC-term	F_1	F_2	F_4	f_8
1	8.5	-4	-2	-1	-.5
2	8.5	-4	-2	-1	.5
3	8.5	-4	-2	1	-.5
4	8.5	-4	-2	1	.5
5	8.5	-4	2	-1	-.5
6	8.5	-4	2	-1	.5
7	8.5	-4	2	1	-.5
8	8.5	-4	2	1	.5
9	8.5	4	-2	-1	-.5
10	8.5	4	-2	-1	.5
11	8.5	4	-2	1	-.5
12	8.5	4	-2	1	.5
13	8.5	4	2	1	.5
14	8.5	4	2	-1	.5
15	8.5	4	2	1	-.5
16	8.5	4	2	1	.5

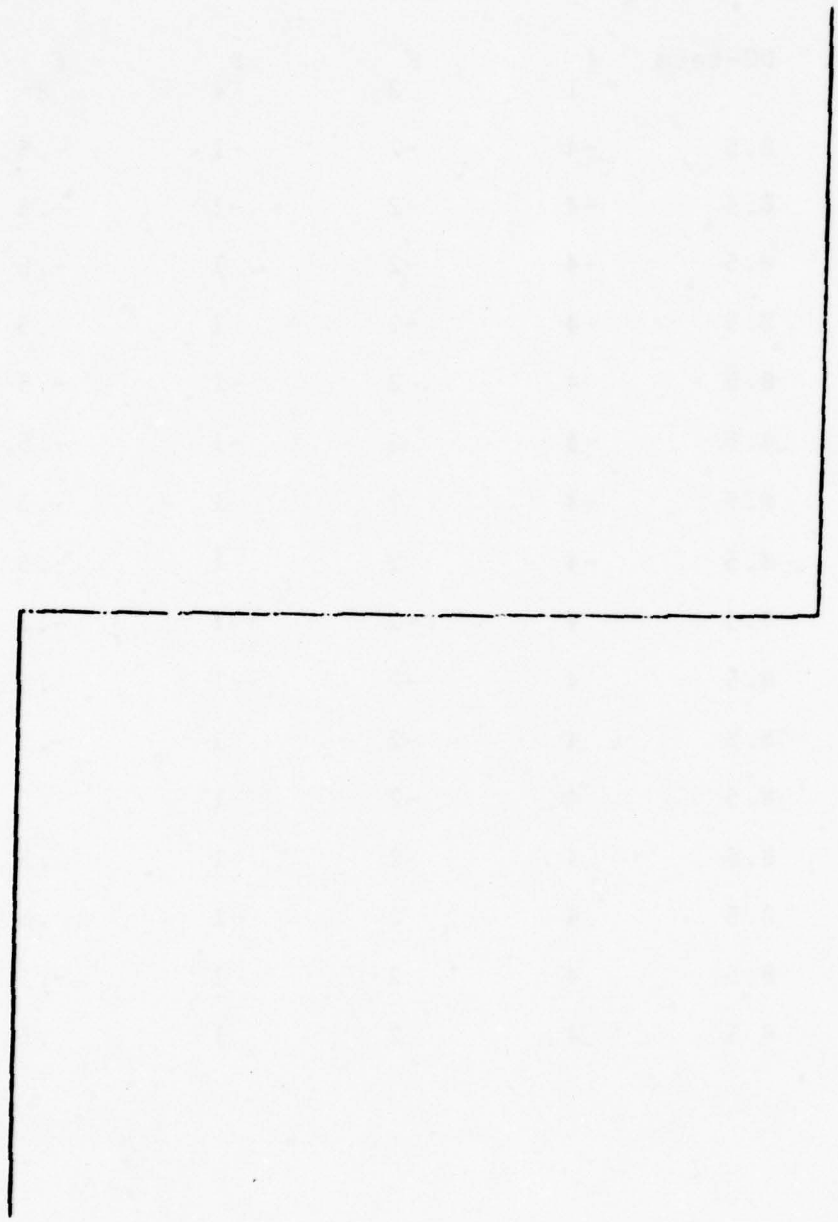

Figure 6. 2 Haar components for sine synthesis.

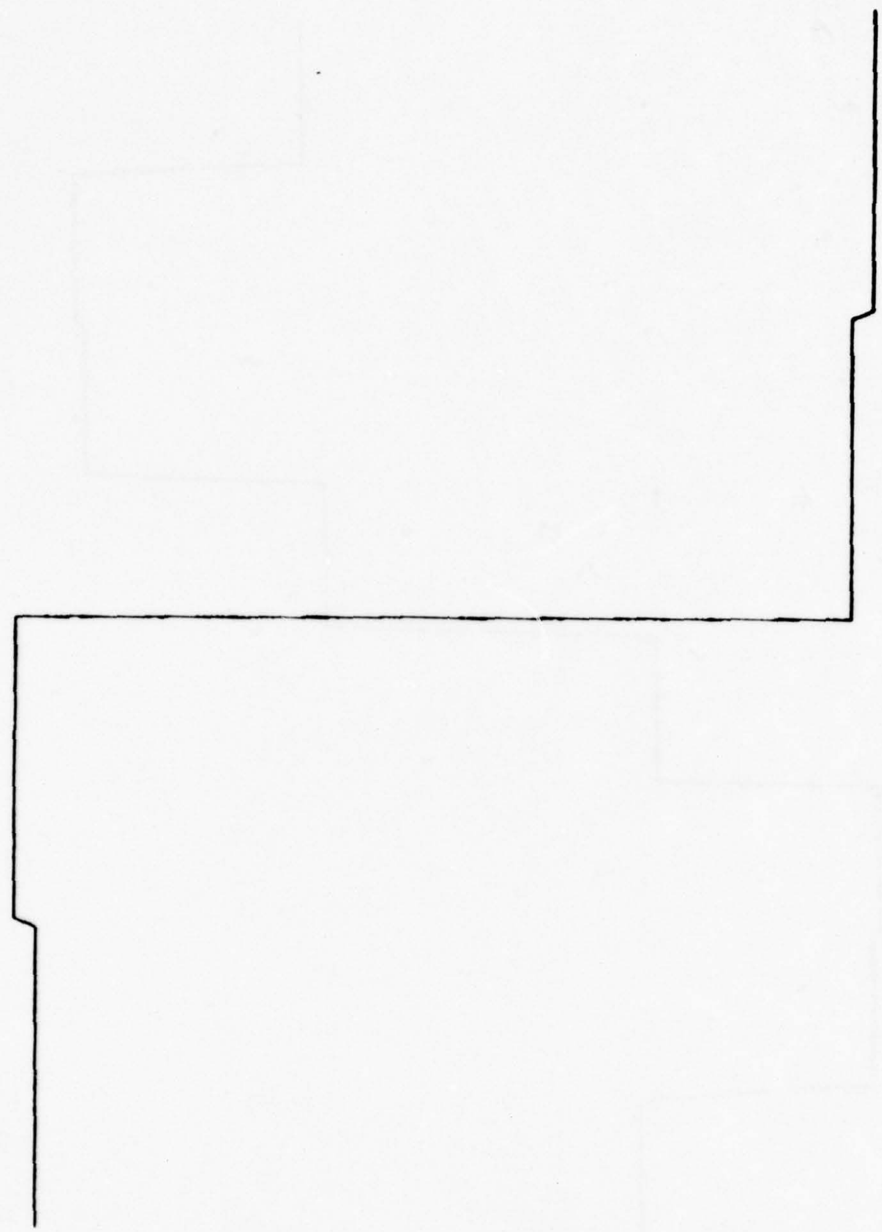

Figure 7.   3 Haar components for sine wave synthesis.

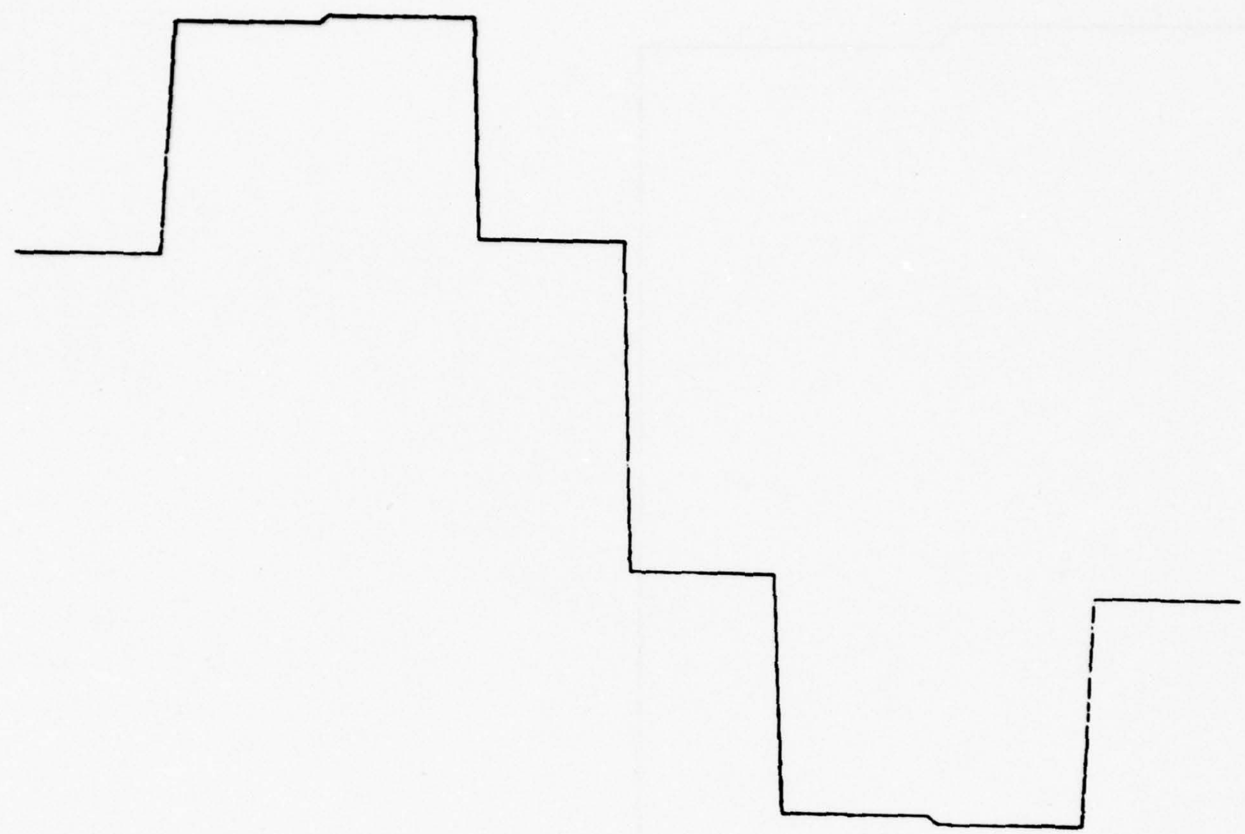

Figure 8. 4 Haar components for sine wave synthesis.

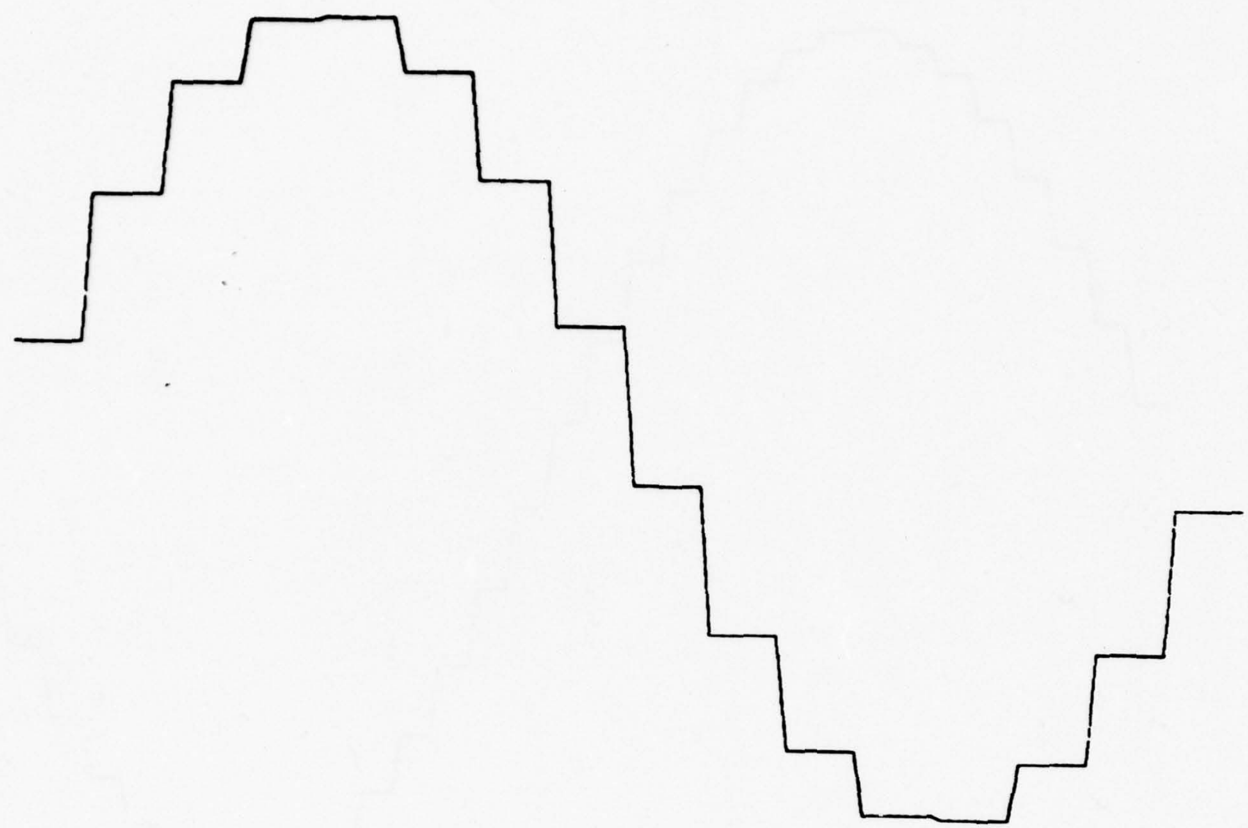

Figure 9. 5 Haar components for sine wave synthesis.

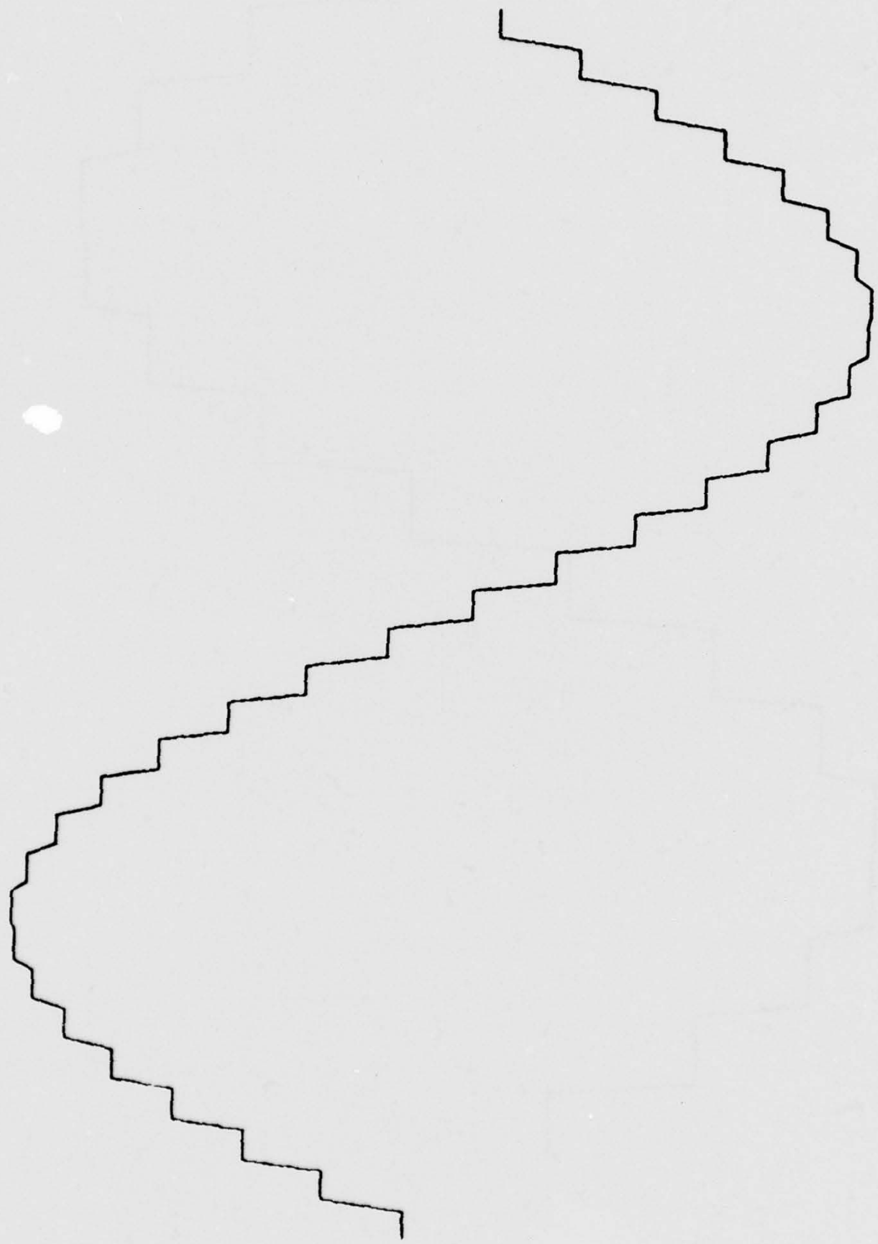

Figure 10. 6 Haar components for sine wave synthesis.

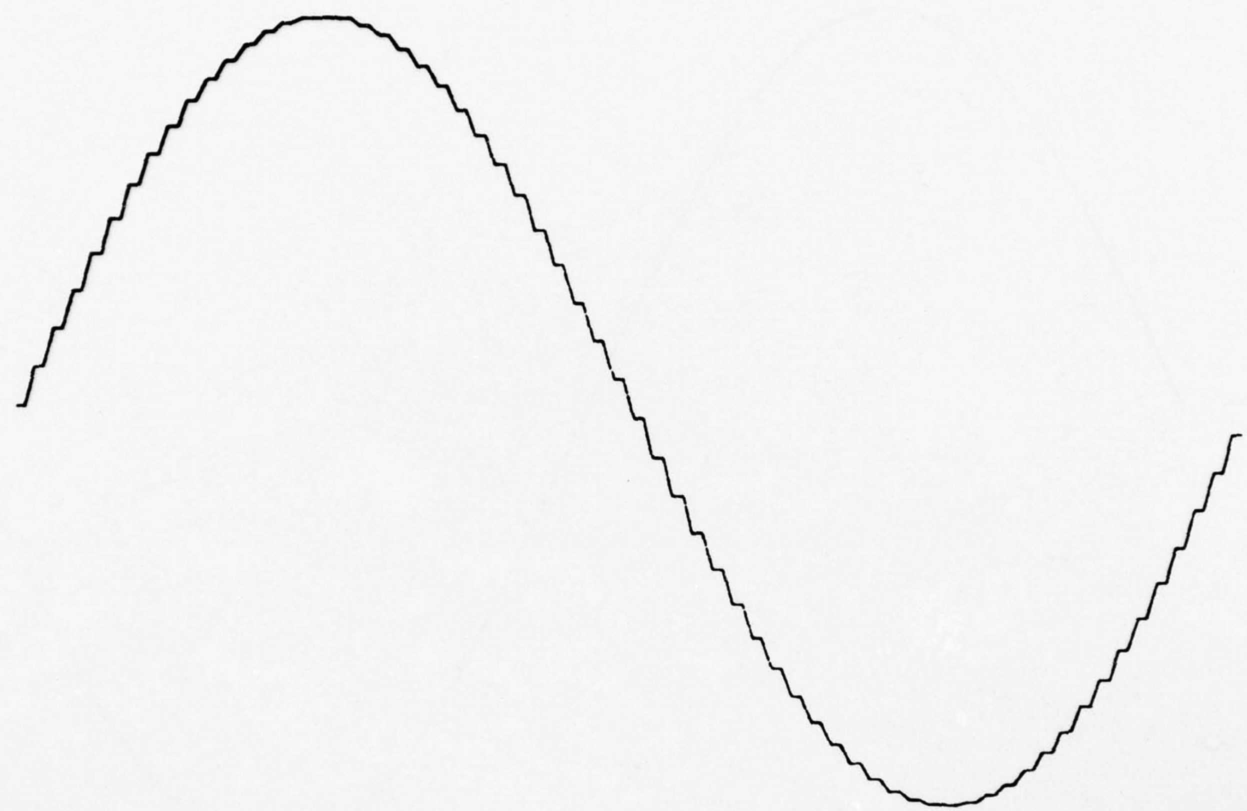

Figure 11. 7 Haar components for sine wave synthesis.

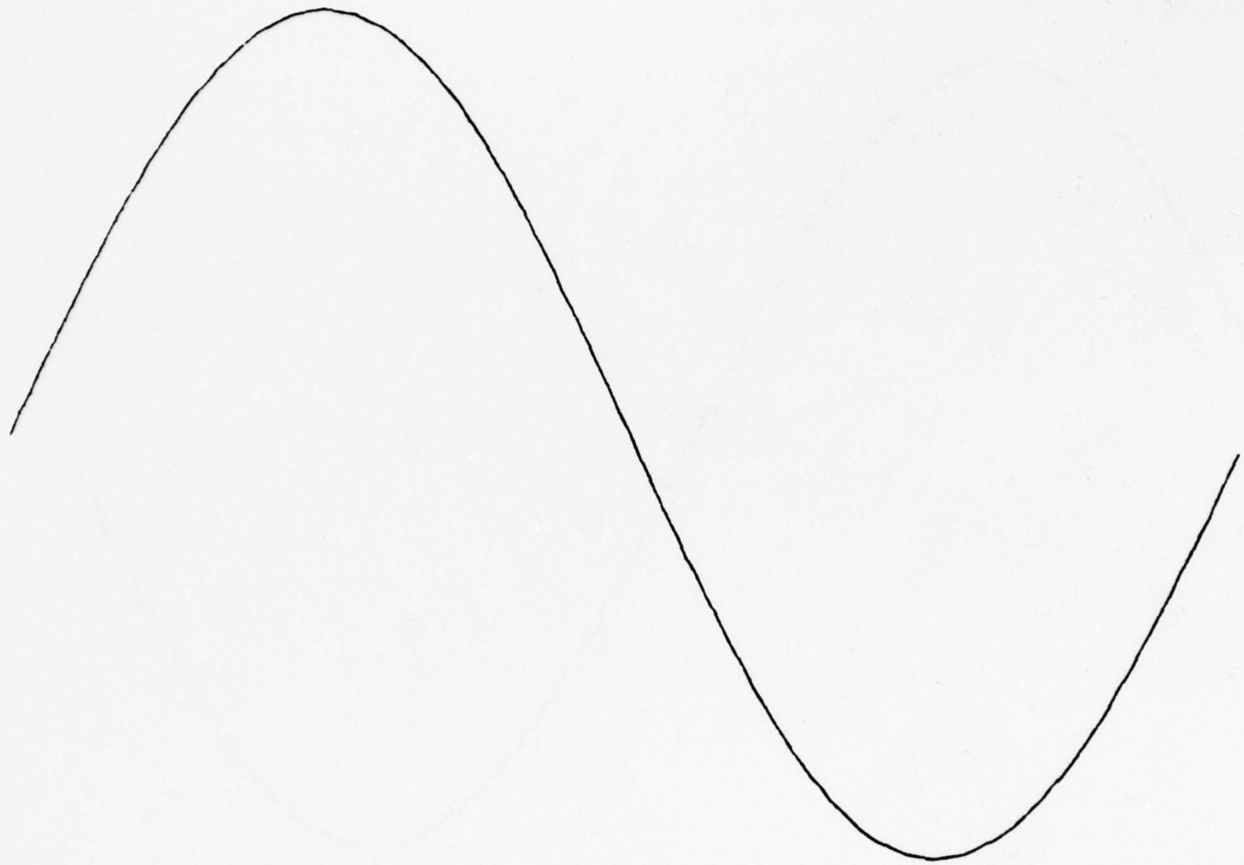

Figure 12. All 8 Haar components: synthesized sine wave.

CONCLUSIONS

A new powerful and versatile tool is now available for digital data analysis. Some of the advantages that the Fast Haar Transform has over the Fast Fourier Transform are:

1. Computations are done between 4 and 5 times faster in terms of the computer time required, thereby realizing an 80 percent savings in computer time and cost.

2. No requirements to generate or store in memory, tables of trigonometric values.

3. Aliasing ambiguities are eliminated.

4. It is optimally suited for pulsed data, and image analysis and enhancement.

Although available in the literature, material on the theory and applications of the Haar Transform is not prolific. Hopefully the presentation of this original work has provided the needed information in a clear and logical manner, and will be of value as an aid in digital data processing applications.

REFERENCES

1. G. Alexits, Convergence Problems of Orthogonal Series, Pergamon, NY, 1961, p 46-72

2. Gary Sivak, "Advanced Video Signal Processing for Image Enhancement," Frankford Arsenal 1976 Technical Symposium, p 95-114

APPENDIX A

PROGRAM HIMAT

```
      PROGRAM HIMAT(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT)
      DIMENSION MAT(16,16)
      DO 1 I = 1,16
      DO 1 J = 1,16
    1 MAT(I,J) = 0
      DO 2 I = 1,16
      MAT(I,1) = 1
      IF(I .LE. 8) MAT(I,2) = 1
    2 IF(I .GT. 8) MAT(I,2) = -1
      DO 3 J = 3,4
      DO 3 I = 1,8
      K1 = (I-1) /4
      K = J-3
      L = 2*(-1)**K1
      I1 = I + 8*K
    3 MAT(I1,J) = L
      DO 4 J = 5,8
      DO 4 I = 1,4
      K1 = (I-1) /2
      K = J-5
      L = 4*(-1)**K1
      I1 = I + 4*K
    4 MAT(I1,J) = L
      DO 5 J = 9,16
      DO 5 I = 1,2
      K1 = I-1
      K = J-9
      L = 8*(-1)**K1
      I1 = I + 2*K
    5 MAT(I1,J) = L
      DO 6 I = 1,16
    6 WRITE (6,500) (MAT(I,J),J=1,16)
  500 FORMAT (12X,16I3)
      READ *,DUMMY
      STOP
      END
```

APPENDIX B

PROGRAM FIG4

```
 PROGRAM FIG4(INPUT,OUTPUT,TAPE61=100,TAPE62=100)
 DIMENSION LETTER(3),ITAB(138),XCO(46),YCO(46),MSG(42)
 DATA ITAB/1HI,1H1,1H ,1HI,1H2,1H ,1HI,1H3,1H ,1HI,1H4,1H ,1HI,1H5,
&1H ,1HI,1H6,1H ,1HI,1H7,1H ,1HI,1H8,1H ,1HI,1H9,1H ,1HI,1H1,1H0,
&1HI,1H1,1H1,1HI,1H1,1H2,1HI,1H1,1H3,1HI,1H1,1H4,1HI,1H1,1H5,
&1HI,1H1,1H6,1HA,1H1,1H ,1HA,1H2,1H ,1HA,1H3,1H ,1HA,1H4,1H ,
&1HA,1H5,1H ,1HA,1H6,1H ,1HA,1H7,1H ,1HA,1H8,1H ,1HB,1H1,1H ,
&1HB,1H2,1H ,1HB,1H3,1H ,1HB,1H4,1H ,1HC,1H1,1H ,1HC,1H2,1H ,
&1HT,1H1,1H ,1HT,1H2,1H ,1HT,1H3,1H ,1HT,1H4,1H ,1HT,1H5,1H ,
&1HT,1H6,1H ,1HT,1H7,1H ,1HT,1H8,1H ,1HT,1H9,1H ,1HT,1H1,1H0,
&1HT,1H1,1H1,1HT,1H1,1H2,1HT,1H1,1H3,1HT,1H1,1H4,1HT,1H1,1H5,
&1HT,1H1,1H6/
 DATA MSG/70,105,103,32,52,2*32,72,2*97,114,32,
&116,114,97,110,115,102,111,114,109,32,100,105,97,103,114,
&97,109,44,32,78,32,61,32,49,54,32,99,97,115,101/
 CALL CONNEC(5LINPUT,0)
 CALL CONNEC(6LOUTPUT,0)
 XLM = .1
 RM = .1
 TM = .1
 BM = .2
 YFIG = 250.
 CALL INITT(30)
 CALL TERM(2,4096)
 CALL DWINDO(0.,4096.,0.,3120.)
 CALL CSIZE(IXSIZ,IYSIZ)
 XNUMLTS=42.
 XFIG=2048.-XNUMLTS/2.*IXSIZ
 DIM = 110.
 DIM2 = DIM / 2.
 XS = 40.
 YS = 24.
 SPACE = 31.
 SLOW = 10.
 PS = 31.
 HDIST = (1. - XLM - RM )*4096./15.
 VDIST = 3120.*(1. - TM - BM ) / 4.
 HMARG = XLM * 4096.
 VSTART = 3120. * (1. - TM )
 ICO = 0
 JLET = -2
 DO 1 IROW = 1,5
 IF ( IROW .NE. 5 ) NCOL = 2**(5 - IROW)
 IF ( IROW .EQ. 5) NCOL = 16
 SKIP = FLOAT(16 / NCOL )
 DO 1 ICOL = 1,NCOL
 JLET = JLET + 3
```

```
      DO 2 ILET = 1,3
    2 LETTER ( ILET ) = ITAB(JLET + ILET - 1)
      ICO = ICO + 1
      XC = HMARG + SKIP*(HDIST/2.+HDIST*(ICOL-1))
      YC = VSTART- FLOAT(IROW-1) * VDIST
      IF ( ICO .EQ. 19)  XC = XC - .5 * DIM
      IF ( ICO .EQ. 20) XC = XC - DIM
      IF ( ICO .EQ. 22) XC = XC + .5*DIM
      IF( ICO .EQ. 23) XC = XC + .25 * DIM
      IF ( ICO .EQ. 25) XC = XC + DIM
      IF ( ICO .EQ. 27) XC = XC - 2. * DIM
      IF(ICO .EQ. 28) XC = XC -.25*DIM
      IF ( ICO .EQ. 29) XC = XC - .25 * DIM
      IF ( ICO .EQ. 30) XC = XC + .25 * DIM
      XCO(ICO) = XC
      YCO(ICO) = YC
    1 CALL DRAW(LETTER,DIM,XC,YC,XS,YS,SPACE,SLOW,PS)
      NCO1 = -32
      NCO2 = 0
      DO 3 I1 = 1,3
      NDRAW = 2**(5-I1)
      NCO1 = NCO1 + 2*NDRAW
      NCO2 = NCO2 + NDRAW
      DO 3 IDRAW = 1,NDRAW
      JDRAW = (IDRAW+1) / 2
      ICO = NCO1 + IDRAW
      JCO = NCO2 + JDRAW
      XL1 = XCO(ICO) - DIM2
      YL1 = YCO(ICO) - DIM2
      XL2 = XCO(JCO)
      YL2 = YCO(JCO) + DIM2
      CALL MOVEA(XL1, YL1)
    3 CALL DRAWA(XL2,YL2)
      NCO1 = -32
      DO 4 I1 = 1,3
      NDRAW = 2**(5-I1)
      NCO1 = NCO1 + 2 * NDRAW
      NCO2 = NDRAW / 2 + 30
      DO 4 IDRAW = 1,NDRAW
      Q = DIM2 * FLOAT((-1)**IDRAW)
      JDRAW = (IDRAW + 1) / 2
      ICO = NCO1 + IDRAW
      JCO = NCO2 + JDRAW
      XL1 = XCO(ICO) + Q
      YL1 = YCO(ICO) - DIM2
      XL2 = XCO(JCO)
      YL2 = YCO(JCO) + DIM2
      CALL MOVEA(XL1, YL1)
    4 CALL DRAWA(XL2, YL2)
      DO 5 IT = 1,2
      DO 5 IC = 1,2
```

```
      ICO = 28 + IC
      JCO = 30 + IT
      K = 1 +  (IT+IC) / 4
      XL1 = XCO(ICO) + FLOAT((-1)**K) * DIM2
      YL1 = YCO(ICO) - DIM2
      XL2 = XCO(JCO)
      YL2 = YCO(JCO) + DIM2
      CALL MOVEA(XL1, YL1)
    5 CALL DRAWA(XL2, YL2)
      CALL CHRSIZ(1)
      CALL MOVEA(XFIG,YFIG)
      CALL ANSTR(42,MSG)
      CALL ANMODE
      READ *,DUMMY
      STOP
      END
      SUBROUTINE DRAW(LETTER,DIM,XC,YC,XS,YS,SPACE,SLOW,PS)
      DIMENSION LETTER(3), X(4),Y(4),ISIGN(2)
      DATA ISIGN/1H+,1H-/
      XMIN = XC - DIM / 2.
      XMAX = XC + DIM / 2.
      YMIN =  YC - DIM / 2.
      YMAX = YC + DIM / 2.
      X(1) = XMIN
      X(2) = XMAX
      X(3) = XMAX
      X(4) = XMIN
      Y(1) = YMIN
      Y(2) = YMIN
      Y(3) = YMAX
      Y(4) = YMAX
      CALL CHRSIZ(3)
      DO 1 I = 1,2
      XL = XC - XS + FLOAT(I-1) * SPACE
      IF(LETTER(3).EQ." ") XL = XL + 10.
      YL = YC - YS - FLOAT(I-1) * SLOW
      CALL MOVEA(XL, YL)
      IF(I.EQ.2) CALL CHRSIZ(4)
    1 CALL A1OUT(I,LETTER(I))
      CALL MOVEA(X(4), Y(4))
      DO 2 I = 1,4
    2 CALL DRAWA(X(I), Y(I))
      DO 3 I = 1,2
      XL = XMIN + FLOAT(I-1)*DIM + FLOAT(I-2)*PS
      IF(I.EQ.2) XL = XL + 15.
      YL = YMIN
      CALL MOVEA(XL, YL)
    3 CALL A1OUT(1,ISIGN(I))
      CALL ANMODE
      RETURN
      END
```

APPENDIX C

PROGRAM HARTST

```
      PROGRAM HARTST(INPUT,OUTPUT)
      DIMENSION X(64),Y(64),Q(126),E(7),VAL(13)
      CALL CONNEC(5LINPUT,0)
      CALL CONNEC(6LOUTPUT,0)
    1 PRINT *,"ENTER THE LOG TO THE BASE 2, THE NUMBER OF POINTS. "
      READ *,NU
      NU1 = NU + 1
      N = 2**NU
      NPC = 2*N
      M = N-2
      DO 2 I = 1,N
    2 X(I) = 0.
      PRINT *,"ENTER ",NU1," MULTIPLIERS. "
      READ *,(E(J),J=1,NU1)
      PRINT *,"FOR SYNTHESIS OR ANALYSIS, TYPE 1 OR 2. "
      READ *,IPICK
      IF(IPICK .NE. 1) GO TO 3
      PRINT *,"ENTER DC-LEVEL. "
      READ *,VAL(1)
      DO 4 I = 1,NU
      IC = 2**(I-1)
      IL = 2*I
      IH = IL + 1
      PRINT *,"FOR ",IC," CY.S OF SPFQ. ",I,", ENTER LOW AND HIGH. "
    4 READ *,VAL(IL),VAL(IH)
      DO 5 I = 1,NU1
      NPC = NPC /2
      DO 5 J = 1,N
      L = 2*I-2 + MOD( (J-1) / NPC ,2)
      IF(L .EQ. 0) L = 1
    5 X(J) = X(J) + VAL(L)
      GO TO 6
    3 PRINT *,"ENTER ",N," INPUTS. "
      READ *,(X(J),J=1,N)
    6 DO 7 I = 1,N
    7 Q(I) = X(I)
      CALL HAAR(Q,E,NU)
      DO 8 I = 1,N
    8 Y(I) = Q(M+I)
      PRINT *,"INPUT     OUTPUT"
      DO 9 I = 1,N
    9 PRINT 430,X(I),Y(I)
  430 FORMAT (1H ,2(F8.3,2X))
      READ *,DUMMY
      IF(DUMMY .EQ. 1) GO TO 1
      STOP
      END
```

```
      SUBROUTINE HAAR(Q,E,NU)
      DIMENSION T(1024),Q(2046),E(11)
      N =2**NU
      NO =N
      ME = NU+2
      IP = (-2)*N
      IR = 0
    5 NO = NO/2
      IP = IP + 4*NO
      IR = IR + 2*NO
      ME = ME-1
      DO 6 J =1,NO
      L = 2*J
      K = L-1
      NOJ = NO+J
      T(NOJ) = E(ME)*(Q(IP+K)-Q(IP+L))
    6 Q(IR+J) =Q(IP+K)+Q(IP+L)
      IF (NO.GT.2) GO TO 5
      T(2) =E(2)*(Q(IR+1)-Q(IR+2))
      T(1) =E(1)*(Q(IR+1)+Q(IR+2))
      Q(1) =(T(1)+T(2))/FLOAT(N)
      Q(2) =(T(1)-T(2))/FLOAT(N)
      IP= -1
      IR = 0
    7 N2 = NO
      X = FLOAT(N2)/FLOAT(N)
      NO = NO*2
      IP = IP + NO/4
      IR = IR + NO/2
      DO 8 J =1,NO
      ND = (J+1)/2
      ID = N2 + ND
      Y = (-1.0)**(J+1)*X
    8 Q(IR+J) = Q(IP+ND)+Y*T(ID)
      IF (NO.LT.N) GO TO 7
      RETURN
      END
```

APPENDIX D

PROGRAM FIG6

```
      PROGRAM FIG6(INPUT,OUTPUT,TAPE61=100,TAPE62=100)
      DIMENSION X(128),Y(128),Q(254),E(8),MSG(50),MSG8(53)
      DATA MSG/70,105,103,2*32,54,46,2*32,50,32,72,2*97,114,32,
     &99,111,109,112,111,110,101,110,116,115,32,102,111,114,32,
     &115,105,110,101,32,119,97,118,101,32,
     &115,121,110,116,104,101,115,105,115/
      DATA MSG8/70,105,103,32,49,50,46,2*32,65,2*108,32,56,32,
     &72,2*97,114,32,99,111,109,112,111,110,101,110,116,115,58,32,
     &115,121,110,116,104,101,115,105,122,101,100,32,
     &115,105,110,101,32,119,97,118,101/
      CALL CONNEC(5LINPUT,0)
      CALL CONNEC(6LOUTPUT,0)
      CALL INITT(30)
      CALL TERM(2,4096)
      XLM = .1
      RM = .1
      TM = .1
      BM = .2
      YFIG = 250.
      XNMSG = 50.
      CALL CSIZE(IXSIZ,IYSIZ)
      XFIG = 2048.-XNMSG/2.*IXSIZ
      PI = 4. * ATAN(1.)
      NU = 7
      NU1 = NU+1
      N = 2**NU
      M = N-2
      NC = 1
      DX = 2. * PI * FLOAT(NC) / FLOAT(N)
      DO 1 NCOMP = 2,8
      DO  2 I=1,NU1
    2 E(I) = 0.0
      DO 3 ICOMP = 1,NCOMP
    3 E(ICOMP) = 1.0
      XMIN = YMIN = YMAX = 0.
      DO 4 I = 1,N
      X(I) = DX * FLOAT(I-1)
    4 Q(I) = SIN(X(I))
      XMAX = X(N)
      CALL HAAR(Q, E, NU)
      DO 5 I = 1,N
      Y(I) = Q(M+I)
      IF ( Y(I) .GT. YMAX) YMAX = Y(I)
    5 IF ( Y(I) .LT. YMIN ) YMIN = Y(I)
      HH = (XMAX-XMIN) / (1. - XLM - RM)
      XMIN = XMIN - HH*XLM
      XMAX = XMAX + HH * RM
```

```
      HV = (YMAX - YMIN ) / (1. - TM - BM)
      YMAX = YMAX + HV * TM
      YMIN = YMIN - HV * BM
      CALL DWINDO(XMIN,XMAX,YMIN,YMAX)
      CALL MOVEA(X(1), Y(1))
      DO 6 I = 2,N
    6 CALL DRAWA(X(I), Y(I))
      IF(NCOMP .EQ. 8) XFIG = XFIG - 2.*IXSIZ
      CALL DWINDO(0.,4096.,0.,3120.)
      CALL MOVEA(XFIG,YFIG)
      IF(NCOMP .EQ. 8) GO TO 7
      IF(NCOMP .EQ. 6) MSG(5) = 49
      IF(NCOMP .EQ. 6) MSG(6) = 47
      IF(NCOMP .NE. 2) MSG(6) = MSG(6) +1
      IF(NCOMP .NE. 2) MSG(10) = MSG(10) + 1
    7 IF(NCOMP .NE. 8) CALL ANSTR(50,MSG)
      IF(NCOMP .EQ. 8) CALL ANSTR(55,MSG8)
      CALL ANMODE
    1 READ 100,ICHAR
  100 FORMAT (A1)
      END
      SUBROUTINE HAAR(Q,E,NU)
      DIMENSION T(1024),Q(2046),E(11)
      N =2**NU
      NO =N
      ME = NU+2
      IP = (-2)*N
      IR = 0
    5 NO = NO/2
      IP = IP + 4*NO
      IR = IR + 2*NO
      ME = ME-1
      DO 6 J =1,NO
      L = 2*J
      K = L-1
      NOJ = NO+J
      T(NOJ) = E(ME)*(Q(IP+K)-Q(IP+L))
    6 Q(IR+J) =Q(IP+K)+Q(IP+L)
      IF (NO.GT.2) GO TO 5
      T(2) =E(2)*(Q(IR+1)-Q(IR+2))
      T(1) =E(1)*(Q(IR+1)+Q(IR+2))
      Q(1) =(T(1)+T(2))/FLOAT(N)
      Q(2) =(T(1)-T(2))/FLOAT(N)
      IP= -1
      IR = 0
    7 N2 = NO
      X = FLOAT(N2)/FLOAT(N)
      NO = NO*2
      IP = IP + NO/4
      IR = IR + NO/2
      DO 8 J =1,NO
```

```
      ND = (J+1)/2
      ID = N2 + ND
      Y = (-1.0)**(J+1)*X
    8 Q(IR+J) = Q(IP+ND)+Y*T(ID)
      IF (NO.LT.N) GO TO 7
      RETURN
      END
```

DISTRIBUTION LIST

Director
US Army Research Office
ATTN: Library
3045 Columbia Pike
Arlington, VA 22204

Director
Department of Defense
Advanced Research and Projects
Agency
Washington, DC 20301

Commander
US Army Materiel Development &
Readiness Command
ATTN: DRCSA-R
DRCSA-PP
Washington, DC 20315

Commander
US Army Aviation Systems Command
12th & Spruce Streets
St. Louis, MO 63166

Commander
US Army Tank-Automotive Research &
Development Command
ATTN: DRDTA, Technical Library
Warren, MI 48090

Commander
US Army Electronics Command &
Devices Laboratory
Ft. Monmouth, NJ 07703

Commander
US Army Electronics Command
Atmospheric Science Laboratory
White Sands Missile Range
ATTN: DRSEL-BL-MS
Mr. Richard Gomez
White Sands, NM 88002

Commander
US Army TECOM
ATTN: AMSTE-TA-A (2)
STEAP-TL
STEAP-DS-LP
Aberdeen Proving Ground, MD 21005

Commander
US Army Harry Diamond Laboratories
ATTN: AMXDO-TD/002
AMXDO-TIB
2800 Powder Mill Road
Adelphi, MD 20783

Commander
US Army ARRADCOM
ATTN: DRDAR-SC LTC (P) A. Larkins
DRDAR-SC Dr. D. A. Gyorog
DRDAR-SCF, L. Berman
DRDAR-SCF-I, J. Lehman
DRDAR-SCF-IE, S. Novak
DRDAR-SCF-IE, G. Sivak (12)
DRDAR-SCP, J. Glennon
DRDAR-MSA, R. Isakower
DRDAR-TSS (5)
Dover, NJ 07801

Defense Documentation Center (12)
Cameron Station
Alexandria, VA 22314

Commander
US Army Armament Materiel and
Readiness Command
ATTN: DRDAR-LEP-L
Rock Island, IL 61299

Director
US Army TRADOC Systems
Analysis Activity
ATTN: ATAA-SL (Tech Lib)
White Sands Missile Range, NM 88002

US Army Materiel Systems
Analysis Activity
ATTN: DRXSY-MP
Aberdeen Proving Ground, MD 21005

Weapon System Concept Team/CSL
ATTN: DRDAR-ACW
Aberdeen Proving Ground, MD 21010

Technical Library
ATTN: DRDAR-CLJ-L
Aberdeen Proving Ground, MD 21005

Technical Library
ATTN: DRDAR-TSB-S
Aberdeen Proving Ground, MD 21010

Technical Library
ATTN: DRDAR-LCB-TL
Benet Weapons Laboratory
Watervliet, NY 12189

Dr. Thomas S. Huang
Institute for Image Technology
1000 North Western Ave.
West Lafayette, Indiana 47906